GPU컴퓨팅

(공영호 교수님)

Assignment2

학번: 2019202103

학과: 컴퓨터정보공학부

이름: 이은비

**Assignment2**

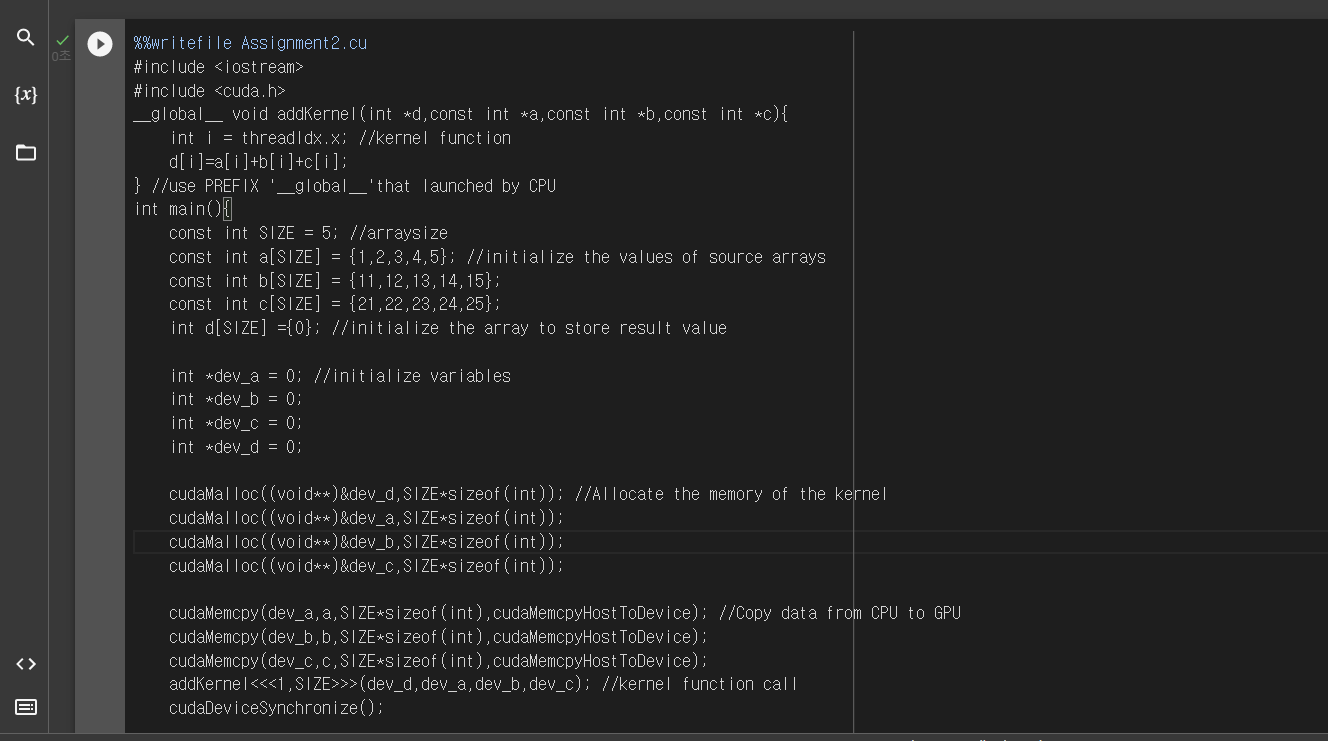
**<Introduction>**

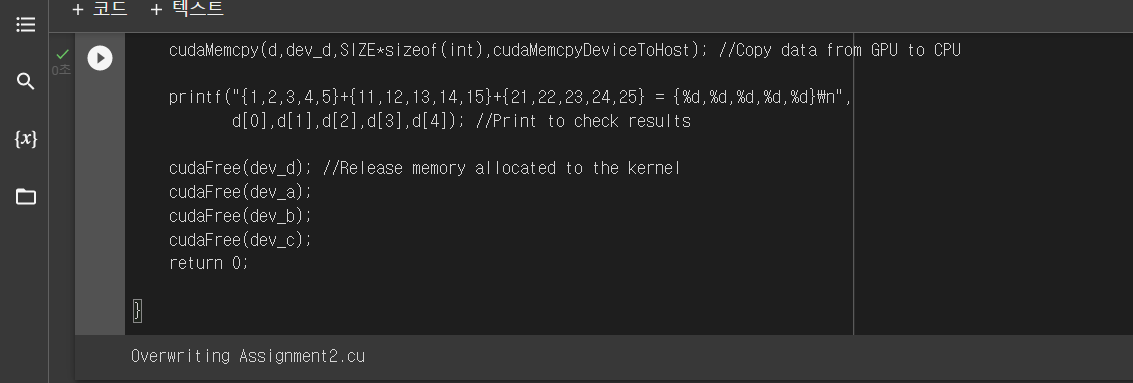
SIZE가 5인 vector addition code를 작성합니다. d[i]= a[I]+b[i]+c[i]와 같은 식을 CUDA-based vector addition을 이용하여 작성합니다. CPU-based vector addition에서는 한개의 CPU에서 순차적으로 진행하여서 SIZE만큼의 시간이 걸리지만 CUDA-based vector addition에서는 수많은 GPU cores를 갖고 동시에 병렬적으로 진행하므로 ‘d[0]=a[0]+b[0]+c[0]’ 식을 진행하는 시간만큼만 소요됩니다. 또한 이 과정에선 CUDA kernel function을 사용하며, cpu에서 launched 되어서 host와 device의 bridge역할을 해주는 PREFIX인 ‘\_\_global\_\_’을 같이 작성합니다.

**<Conclusion>**

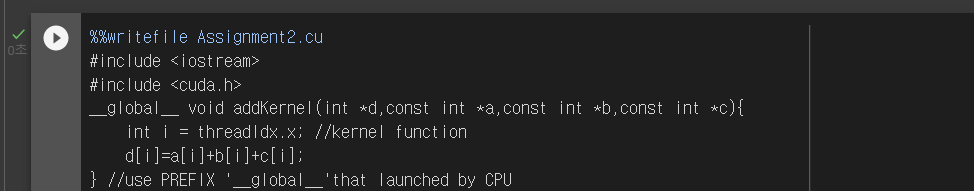
**[Assignment2.cu의 코드내용 및 결과화면]**

**(전체 코드내용)**



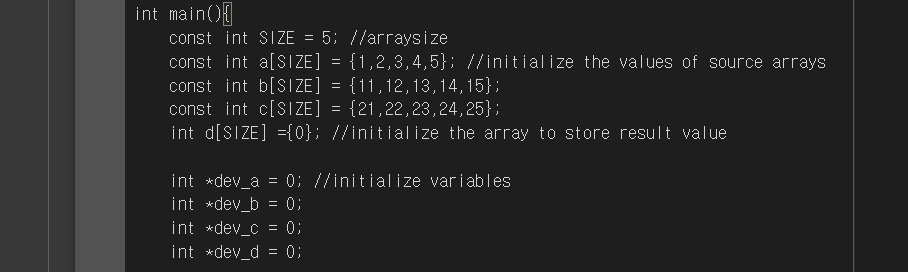


(부분 코드1)



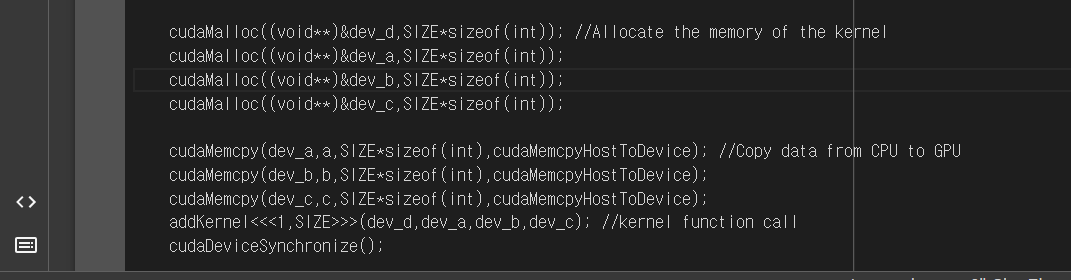
CUDA kernel function을 준비합니다. 이때 \_\_global\_\_은 CPU에서 launched 되어서 GPU에서 삼수의 내용을 수행 할 수 있도록 합니다. 함수의 이름은 addKernel이며 인수는 CPU의 변수이고, i=threadIdx.x; 에서 threadIdx.x는 kernel의 built-in변수로 GPU core마다의 id를 i에 할당합니다. 또한 할당된 값으로 d[i]=a[i]+b[i]+c[i];식을 i=0~SIZE까지 동시에 수행합니다.

(부분 코드2)



Souce arrays(a,b,c)의 값들을 임의의 값으로 초기화 합니다. 그 결과값을 진행할 변수 d도 0으로 초기화 하며, addKernel function을 수행하기 위한 dev\_a,dev\_b,dev\_c,dev\_d도 초기화합니다.

(부분코드3)

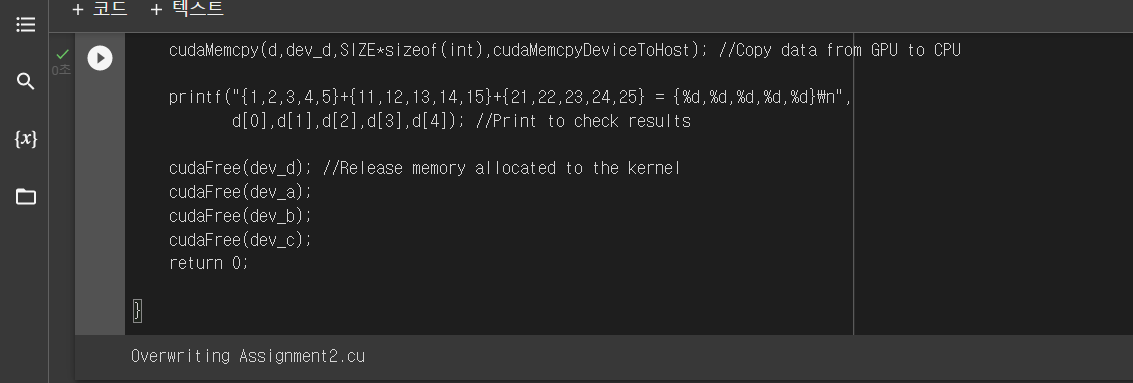


cudaMalloc function을 이용하여 Kernel의 메모리를 할당합니다. 할당되는 사이즈는 전달 받을 사이즈와 같습니다.

cudaMemcpy function을 이용하여 host의 data를 device로 copy합니다. a,b,c배열의 값을 각각dev\_a,dav\_b,dev\_c,로 copy합니다

addKerenl<<<1,SIZE>>>(dev\_d, dev\_a,dav\_b,dev\_c); 1부터 SIZE까지 앞서 prepare된 함수를 호출합니다.

(부분코드4)

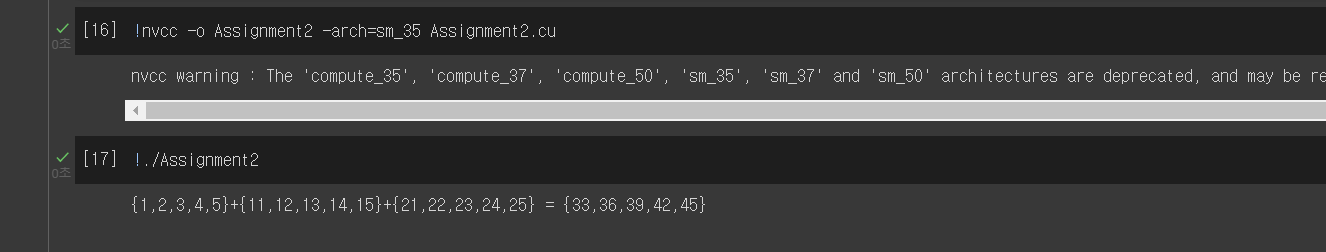


cudaMemcpy를 통해 수행된 kernelfunction의 결과 값이 저장된 device의 dev\_d의 data를 host의 d로 copy합니다.

Printf함수로 수행된 결과값을 출력합니다.

cudaFree();함수로 앞서 cudaMalloc으로 할당되었던 변수의 메모리를 해제 합니다.

(결과화면)



!nvcc -o Assignment2 -arch\_35 Assignment2.cu

앞서 작성되었던 Assignment2.cu파일을 Assignment2라는 이름의 실행파일로 컴파일합니다.

!./Assignment2

실행파일을 실행시켜서 printf로 작성된 문장의 결과가 출력됩니다.